FROM OPTICAL FLOW TO DENSE LONG TERM CORRESPONDENCES

Tomás Crivelli, Pierre-Henri Conze, Philippe Robert and Patrick Pérez

Technicolor, Rennes, France.

ABSTRACT

Dense point matching and tracking in image sequences is an open issue with implications in several domains, from content analysis to video editing. We observe that for long term dense point matching, some regions of the image are better matched by concatenation of consecutive motion vectors, while for others a direct long term matching is preferred. We propose a method to optimally estimate the correspondence of a point w.r.t. a reference image from a set of input motion estimations over different temporal intervals. Results on texture insertion by point tracking in the context of video editing are presented and compared with a state-of-the-art approach.

Index Terms— dense point matching and tracking, optical flow, video editing.

1. INTRODUCTION

The problem of point and patch tracking is a widely studied and still open issue with implications in a broad area of computer vision and image processing [1,2,3,4,5]. On one side and among others, applications such as object tracking, structure from motion, motion clustering and segmentation, and scene classification may benefit from a set of point trajectories by analyzing an associated feature space. In this case, usually a *sparse* or *semi-sparse* [3] set of meaningful points needs to be tracked, and indeed, those points that carry important information about the structure of the scene are more easily tracked. Recent approaches as those presented in [1,2] are examples of the importance of long-term motion cues for spatio-temporal video segmentation.

On the other side, applications related to video processing such as augmented reality, texture insertion, scene interpolation, view synthesis, video inpainting and 2D-to-3D conversion eventually require determining a *dense* set of trajectories or point correspondences that permit to propagate large amounts of information (color, disparity, depth, position, etc.) across the sequence. Dense motion information is well represented by optical flow fields and points can be simply propagated through time by accumulation of the motion vectors. That is why state-of-the-art methods have built on top of optical flow methods for dense point tracking [1,3,5].

Our approach is similar as we exploit a set of input motion fields computed independently, which we call *elementary motion fields*. This set, however, is composed by motion fields obtained with different estimation steps, i.e., time intervals between pairs of images. We have observed that for long term dense point matching, some regions of the image are better matched by concatenation of instantaneous motion vectors, while for others a direct long term matching is preferred.

The contribution of this work is two-fold: first we propose a novel sequential method of accumulating elementary motion fields to produce a long term matching; second, we show how to optimally combine different motion estimation steps in order to decide for the best point correspondence between two images. We present results in the context of video editing for automatic logo insertion by point tracking. Comparisons w.r.t. a state-of-the-art approach are given.

2. MULTI-STEP POINT MATCHING

2.1. Sequential displacement field construction

Consider an image sequence $\{I_n\}_{n:0\dots N}$ and let the last image I_N be the *reference image*. Our objective is to compute the displacement vector at each location of each image w.r.t. the reference, i.e. $\boldsymbol{d}_{n,N}(\boldsymbol{x}_n)$, for each *n,* where $\boldsymbol{x}_n$ belongs to the image grid Ω. For the time being, we only assume that the elementary motion fields, $\boldsymbol{d}_{n,n+1}$, $n = 0 \dots N-1$, computed between pairs of consecutive frames are available as input information.

In previous point tracking approaches based on optical flow [1,3,5], a simple 1st-order Euler integration is conducted as follows: 1) take a starting grid point $\boldsymbol{x}_n \in \Omega$ in I_n, 2) for $m = n, n+1 \dots N-1$ obtain iteratively

$$\boldsymbol{x}_{m+1} = \boldsymbol{x}_m + \boldsymbol{d}_{m,m+1}(\boldsymbol{x}_m), \qquad (1)$$

3) repeat for each $\boldsymbol{x}_n$. This gives an estimate of the positions of the points at time N, by forwards concatenation of elementary motion fields. This simple scheme can then be combined with a more sophisticated global formulation for track estimation [3].

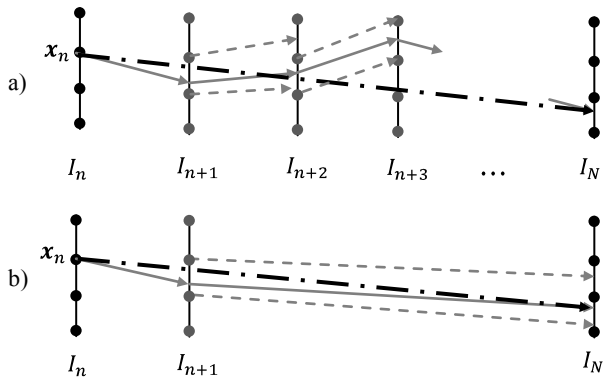


Fig. 1: Estimation of $\boldsymbol{d}_{n,N}(\boldsymbol{x}_n)$. a) Scheme corresponding to (1): elementary motion vectors are interpolated and then accumulated. b) Scheme corresponding to our method (2): a previously estimated long term displacement is interpolated and then accumulated with an elementary motion vector. Dashed arrows indicate the displacement vectors at grid locations used for interpolation.

Our approach is based on a different strategy that runs backwards and aims at computing $\boldsymbol{d}_{n,N}(\boldsymbol{x}_n)$ while exploiting the elementary motion fields. It is given by the following iteration:

$$\boldsymbol{d}_{n,N}(\boldsymbol{x}_n) = \boldsymbol{d}_{n,n+1}(\boldsymbol{x}_n) + \boldsymbol{d}_{n+1,N}\left(\boldsymbol{x}_n + \boldsymbol{d}_{n,n+1}(\boldsymbol{x}_n)\right), \quad (2)$$

for each grid location $\boldsymbol{x}_n$ in I_n. That is, the current *long-term* displacement field $\boldsymbol{d}_{n,N}$ is obtained by concatenation of the previously computed *long-term* field $\boldsymbol{d}_{n+1,N}$ and an elementary motion field $\boldsymbol{d}_{n,n+1}$.

Note the difference between (1) and (2). Starting from the grid point $\boldsymbol{x}_n$ at image I_n, and its elementary displacement $\boldsymbol{d}_{n,n+1}(\boldsymbol{x}_n)$, one computes $\boldsymbol{x}_n + \boldsymbol{d}_{n,n+1}(\boldsymbol{x}_n)$. Then, in the former approach (Eq. 1), one interpolates the velocity $\boldsymbol{d}_{n+1,n+2}\left(\boldsymbol{x}_n + \boldsymbol{d}_{n,n+1}(\boldsymbol{x}_n)\right)$ in I_{n+1} (e.g. by bilinear interpolation), and continues accumulating elementary motion vectors in the forward direction (Fig. 1a). In the second approach, the interpolation is applied once on the long term motion field $\boldsymbol{d}_{n+1,N}\left(\boldsymbol{x}_n + \boldsymbol{d}_{n,n+1}(\boldsymbol{x}_n)\right)$ directly between instants $n+1$ and N. This procedure implies that $\boldsymbol{d}_{n+1,N}$ in (2) is available from the previous iteration (Fig. 1b). The result is that we sequentially compute the dense displacement maps $\boldsymbol{d}_{n,N}$ backwards, for every frame n with respect to the reference frame N.

In order to obtain the correspondence between all pixels of all images w.r.t. the reference, it is easy to see that for the standard method the complexity is $O(N^2P)$ while for the proposed method it is $O(NP)$, where P is the number of pixels for a single image. Besides a higher efficiency, it also appears that this approach is more accurate.

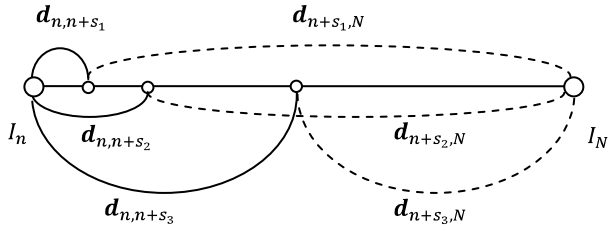


Fig. 2: Multi-step point correspondence. For a given point, the displacement from frames *n* to *N* can be obtained through different paths according to the input elementary motion fields (solid lines) and the previously estimated long-term displacements (dashed lines).

2.2. Multi-step flow

Now we exploit the previous strategy for defining an optimal and sequential way of combining elementary motion fields estimated with different frame *steps* (i.e. the time interval between two frames) in order to obtain an improved and dense displacement map. The reasoning is based on the following. We want to compute $\boldsymbol{d}_{n,N}(\boldsymbol{x}_n)$. Suppose that for a set of Q_n frame steps at instant n, say $S_n = \{s_1, s_2, s_3, \dots, s_{Q_n}\} \subset \{1, \dots, N-n\}$, the set of corresponding motion fields $\left\{\boldsymbol{d}_{n,n+s_1}, \boldsymbol{d}_{n,n+s_2}, \dots, \boldsymbol{d}_{n,n+s_{Q_n}}\right\}$ is available. For each $s_k \in S_n$ we write

$$\boldsymbol{d}^k_{n,N}(\boldsymbol{x}_n) = \boldsymbol{d}_{n,n+s_k}(\boldsymbol{x}_n) + \boldsymbol{d}_{n+s_k,N}\left(\boldsymbol{x}_n + \boldsymbol{d}_{n,n+s_k}(\boldsymbol{x}_n)\right). \quad (3)$$

In this manner we generate different candidate displacements or *paths* (Fig. 2) among which we aim at deciding the optimal for each location $\boldsymbol{x}_n$. With $Q_n = 1\ \forall n$ and $s_1 = 1$ it reduces to (2). This scheme is somewhat related to that presented in [7] for computing a single optical flow field between two given images, where several candidate solutions are fused on the basis of a global optimization framework.

2.3. Optimal path selection

Let us recall the setting so far: we want to compute $\boldsymbol{d}_{n,N}(\boldsymbol{x}_n)$; we have defined and computed the Q_n candidates $\boldsymbol{d}^k_{n,N}(\boldsymbol{x}_n)$ for every point $\boldsymbol{x}_n$ in image I_n and now the best one has to be selected at each location. For that sake, we need to define an optimality criterion and an optimization strategy. We first define the function $C_{n,N}(\boldsymbol{x}_n, \boldsymbol{d})$ as a matching cost between location $\boldsymbol{x}_n$ in image I_n and location $\boldsymbol{x}_n + \boldsymbol{d}$ in I_N. It can be arbitrarily constructed so as to exploit different spatio-temporal image cues in order to evaluate the goodness of the match. For the results presented here, it is defined in section 3. Deciding for each location $\boldsymbol{x}_n$ independently by selecting k such that $C_{n,N}\left(\boldsymbol{x}_n, \boldsymbol{d}^k_{n,N}(\boldsymbol{x}_n)\right)$ is minimized may result in the

introduction of an undesired noise in the final motion field, as neighboring image points will be frequently assigned with motion values computed with different values of k. Moreover, the proposed cost may not be robust enough. Thus, we improve the result by embedding it together with a spatial Potts-like regularization process. Let $\boldsymbol{K} = \{k_{\boldsymbol{x}}\}$ be a full labeling of the image grid, where each label $k_{\boldsymbol{x}}$ indicates one of the available candidate paths. We introduce the energy function:

$$E_{n,N}(\boldsymbol{K}) = \sum_{\boldsymbol{x}} C_{n,N}\left(\boldsymbol{x}, \boldsymbol{d}_{n,N}^{k_x}(\boldsymbol{x})\right) - \sum_{<\boldsymbol{x},\boldsymbol{y}>} \alpha \cdot \delta_{k_x = k_y}, \quad (4)$$

where $<\boldsymbol{x},\boldsymbol{y}>$ is a pair of neighboring image locations according to the 4-point connected neighborhood, $\delta_{k_x=k_y}$ is the Kronecker delta and α the spatial regularization parameter defined in (6)**.** We obtain the optimal $\boldsymbol{K}^*$ by applying a graph-cut-based minimization [6]. This in turn gives the optimal long-term correspondence field $\boldsymbol{d}_{n,N}^*(\boldsymbol{x}) = \boldsymbol{d}_{n,N}^{k_{\boldsymbol{x}}^*}(\boldsymbol{x})$.

3. DENSE POINT TRACKING AND VIDEO EDITING

3.1. Sequential forward-backward processing

The multi-step algorithm was described on the basis of a set of *forward* motion fields as inputs. The result is a forward correspondence vector for each point of each image before N. This reasoning is especially useful for video editing tasks, e.g. for the consistent insertion of graphics elements such as logos. Basically, one is able to edit frame N, and then propagate the modified values to the preceding frames using the estimated correspondence fields. Analogously, using backward motion fields as inputs one can readily consider I_0 as the reference image instead. Note that in applications where one needs to track points originated in the reference image (as opposed to track points all the way to the reference frame), it is better to apply the iteration in a different manner. In order to track each pixel $\boldsymbol{x}_N$ in I_N in the backward direction we write:

$$\boldsymbol{d}_{N,n}^{k}(\boldsymbol{x}_N) = \boldsymbol{d}_{N,n+s_k}(\boldsymbol{x}_N) + \boldsymbol{d}_{n+s_k,n}\left(\boldsymbol{x}_N + \boldsymbol{d}_{N,n+s_k}(\boldsymbol{x}_N)\right), \quad (5)$$

so that for each starting location we can compute the position at precedent frames. Similarly, using forward motion fields, we can track all the points from image I_0 in the forward direction. It is worth to say that combining these different variations of the algorithm, one can track and match (forward and backward) all the pixels of a reference image arbitrarily picked from within the sequence.

3.2. Input data and parameter selection

For the main results presented in this paper, we consider a sequence of 1920x1080 HD video frames with $N = 100$ (*AmeliaRetro*, courtesy of Dolby®). To each $s \in \bigcup_{n=0}^{N-1} S_n$, we associate a *leap* l such that if $n \bmod l = 0$ then $s \in S_n$. We then have a set of input motion fields that we pre-estimate by an adapted 2D version of the 1D disparity estimator described in [8] and for the set of (s,l) pairs $\{(1,1),\ (2,1),\ (5,5),\ (10,10),\ (20,10),\ (30,10),\ (50,10),\ (80,10)\}$.

We also define $C_{n,N}(\boldsymbol{x}_n, \boldsymbol{d})$ in (4) as the normalized sum of squared differences of pixel color values between image windows of size 5x5. Though this matching criterion may not be invariant to possible scale changes, illumination variations, large deformations and motion discontinuities, we have decided to keep it simple, as it permits to better observe the benefits of the multi-step approach. Meanwhile, the parameter α equals

$$\alpha \equiv \alpha_{xy}^n = e^{-\frac{\|c_x^n - c_y^n\|^2}{\sigma^2}}, \quad (6)$$

with $\boldsymbol{c}_{\boldsymbol{x}}^n, \boldsymbol{c}_{\boldsymbol{y}}^n$ the 3-channel color vectors at locations $\boldsymbol{x}$ and $\boldsymbol{y}$, for image n, respectively. The value $\sigma^2 = 3 \cdot (100)^2$ is set manually or can be estimated locally from the color images. This enforces smoothness of the labels assigned to nearby pixels with similar color.

3.3. Results

Having manually inserted a logo at frame $N = 100$ (Fig. 3a), we modify each image I_n by copying the color values from I_N according to the displacement field $\boldsymbol{d}_{n,N}^*(\boldsymbol{x}_n)$ (Fig 3b). We only edit the image at those pixels that fall inside a predefined insertion mask $R_N \subseteq \Omega$ in I_N, i.e., $\boldsymbol{x}_n$ such that $\boldsymbol{x}_n + \boldsymbol{d}_{n,N}^*(\boldsymbol{x}_n) \in R_N$. For the rest, we leave the pixel unmodified. Note that our method notably reduces deformation, compared to the accumulation (1) of the same optical flow fields obtained with [8] as well as those given by [5] (implementation provided by the authors). In Fig. 4 we evaluate the accuracy of each method by propagating a region of the reference image to the whole sequence and computing the color PSNR w.r.t. the original input images at each instant n. Our method shows a clear improvement at all frames especially for long term correspondences. The poor results obtained for Brox *et al*. [5] is a consequence of small spurious defects in the motion fields between successive frames, which are propagated to the final long-term displacement estimation. Finally, the ability of our method to combine optical flow fields estimated with different frame distances (*steps*) allows us to handle complex situations as, for example, logo insertion in the presence of occlusions/disocclusions. As depicted in Fig. 5, we can observe that some regions of the logo may be occluded at some instant of the sequence but they can be recovered when they reappear thanks to the long term matching. Moreover, pixels are modified only on disoccluded areas, as one would expect.

4. CONCLUSION

Dense point correspondences over time can be notably enhanced by considering multi-step flow fields. We have described a method to optimally combine several flow estimations also exploiting a new motion accumulation strategy. In fact, any elementary optical flow method can be leveraged with this scheme.

5. REFERENCES

[1] Brox, M. and Malik, J. "Object segmentation by long term analysis of point trajectories". In *Proc. ECCV, 2010.*

[2] Fradet, M; Robert, P.; Pérez, P. "Clustering point trajectories with various life-spans". In *Proc. IEEE CVMP, 2011.*

[3] Sand, P. and Teller, S. "Particle Video: Long-Range Motion Estimation Using Point Trajectories", *IJCV*, 80(1), 72-91, 2008.

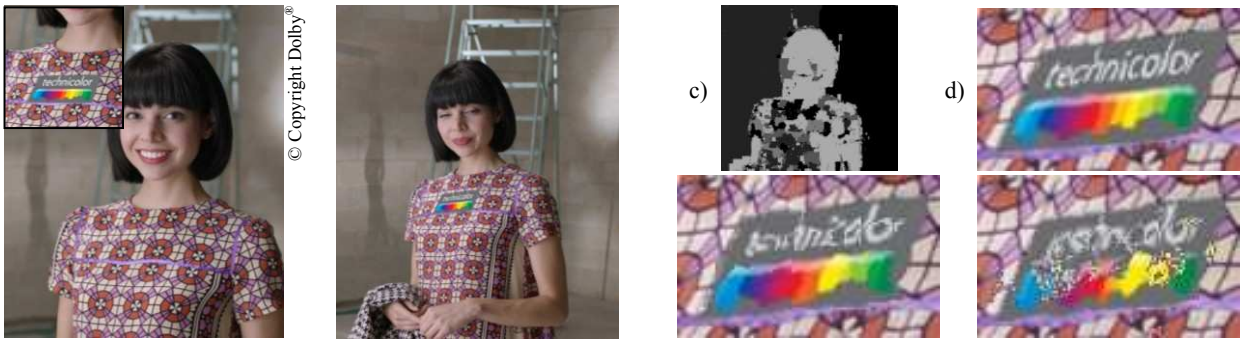
[4] Buchanan, A.; Fitzgibbon, A., "Combining local and global motion models for feature point tracking". In *Proc. IEEE CVPR,* 2007.

[5] Sundaram, N.; Brox, T.; Keutzer, K. "Dense point trajectories by GPU-accelerated large displacement optical flow". In *Proc. ECCV*, 2010.

[6] Boykov, Y.; Veksler, O.; Zabih, R. "Efficient Approximate Energy Minimization via Graph Cuts", *IEEE Trans. Pattern Anal. Mach. Intell*, 20(12), 1222-1239, Nov, 2001

[7] Lempitsky, V.; Roth, S.; Rother, C., "FusionFlow: Discrete-continuous optimization for optical flow estimation". In *Proc. IEEE CVPR*, 2008.

[8] Robert. P; Thébault. C; Conze, P.-H. "Disparity-compensated view synthesis for s3D content correction", In *Proc. SPIE SDA XXIII*, 2012.



a) Logo manually inserted in I_{100}. b) Insertion in I_{10} for our method. e) f)

Fig. 3: Logo insertion. a-b) See image captions. c) Computed path label map $\boldsymbol{K}^*$ for frame I_{60}. Logo propagated at frame $n = 10$ for d) Multi-step flow. e) Accumulation of optical flows from [8]. f) Accumulation of optical flows from Brox *et al*. [5].

Color PSNR (dB). Multi-step Flow. Accum. of OFs from [8]. Accum. of OFs from Brox *et al*.

Fig. 4: Image registration error through time. Left**:** PSNR (dB) between original frame and reconstructed frame for each instant n, by propagating a region of I_{100} to the past. The depicted images illustrate the reconstruction provided by every method at frame $n = 0$.

a) b) c) d) e)

Fig. 5: Logo insertion in the presence of occlusions. a) Original image I_{95}. b) Logo inserted at $n = 95$. c-e) Frames $n = 100,103,105$ of the resulting sequence obtained with our method. In each frame, pixel colors are only modified on non-occluded regions and points that reappear after an occlusion can be consistently matched with the reference image without losing them.